Journal of
Computational and Theoretical Nanoscience
Vol. 2, 473–491, 2005

Computational Intelligence, Bioinformatics and Computational Biology: A Brief Overview of Methods, Problems and Perspectives

Nikola Kasabov,[1,2,*] Igor A. Sidorov,[3] and Dimiter S. Dimitrov[3]

[1] Knowledge Engineering and Discovery Research Institute and School of Computer and Information Sciences, Auckland University of Technology, Auckland 1020, New Zealand
[2] Pacific Edge Biotechnology Ltd, University of Otago Center for Innovation, Dunedin, New Zealand
[3] Center for Cancer Research Nanobiology Program, CCR, NCI-Frederick, National Institutes of Health, Frederick, MD 21702, USA

The paper is an overview of methods of computational intelligence (CI) used in the area of Bioinformatics (BI) for the purpose of advancing the area Computational Biology (CB) and facilitating discoveries from biological data. CI is the area of developing generic intelligent information processing methods and systems with wider applications, one of them being Bioinformatics. CI adopts many principles from Biology, thus offering suitable methods and tools for BI. While CB aims at understanding the biology principles through their computational modeling, BI is aiming at the use and the development of new information methods and systems to enhance the storage, the analysis, modeling, and discovery from biological data. The synergism between the three disciplines, their methodologies, problems, and some current solutions are review in the paper. Some new methods and experimental results are introduced, such as feature and model optimization with genetic algorithms applied on gene expression data.

Keywords: Computational Intelligence, Neural Networks, Bioinformatics, Computational Biology, Genomics, Proteomics, System Biology, Gene Expression Profiling, Gene Regulatory Networks.

CONTENTS

*Author to whom correspondence should be addressed.

1. INTRODUCTION

With the completion of the sequence draft of the human genome and the genomes of other species (more to be sequenced during this century) the task is now to be able to process this vast amount of ever growing dynamic information and to create intelligent systems for prediction and knowledge discovery at different levels of life, from cells to whole organisms and species.

The central dogma of the molecular biology is that the DNA (Deoxyribonucleic Acid) present in the nucleus of each cell of an organism is transcribed into RNA, which is translated into proteins. Genes are complex molecular structures that cause dynamic transformation of one substance into another during the whole life of an individual, as well as the life of the human population over many generations.

Even the static information about a particular gene is very difficult to understand (see the GenBank database www.genebank.com). When genes are "in action," the dynamics of the processes in which a single gene is involved are thousand times more complex, as this gene interacts with many other genes, proteins, and is influenced by many environmental and developmental factors.

Modeling these interactions, learning about them and extracting knowledge, is a major goal for the scientific area of Bioinformatics. Bioinformatics (BI) is concerned with the application and the development of the methods of information sciences for the storage, the analysis, modeling, and knowledge discovery from biological data.

The area of Computational Biology (CB) aims at understanding biology through their computational modeling. CB is "closer" to biology, while seemingly BI is "closer" to the computer and information sciences.

Computational Intelligence (CI), which is part of computer and information sciences, is concerned with the development of generic intelligent information processing methods and systems with wider applications, one of them being BI. CI adopts many principles from biology, thus offering suitable methods and tools for BI and CB.

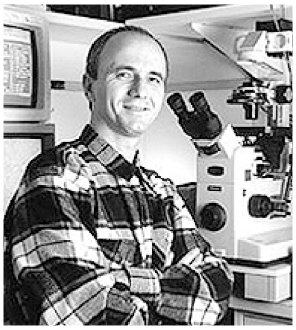
The synergism between the three disciplines, their methodologies, problems, and some current solutions are reviewed in the paper. Section 2 offers a brief review of the most popular methods of CI. Section 3 gives a background information in molecular biology and offers a review of problems in CB and BI along with some solutions using CI methods. Section 4 discusses the impact of the research in this area on medicine and nano-technology.

2. METHODS OF COMPUTATIONAL INTELLIGENCE: A BRIEF OVERVIEW

CI is the area of developing generic intelligent information processing methods and systems with wider applications. CI methods, in its majority, are inspired by the human

Prof. Nikola Kasabov is the Founding Director and the Chief Scientist of the Knowledge Engineering and Discovery Research Institute KEDRI, Auckland. He holds a Chair of Knowledge Engineering at the School of Computer and Information Sciences at Auckland University of Technology. He is a Fellow of the Royal Society of New Zealand, Fellow of the New Zealand Computer Society, Senior Member of IEEE, member of the TC12 of IFIP. He holds M.Sc. and Ph.D. from the Technical University of Sofia, Bulgaria. His main research interests are in the areas of: intelligent information systems, soft computing, neuro-computing, bioinformatics, brain study, speech and image processing, novel methods for data mining and knowledge discovery. He has published more than 300 publications that include 15 books, 80 journal papers, 50 book chapters, 25 patents and numerous conference papers. He has extensive academic experience at various academic and research organisations: University of Otago, New Zealand; University of Trento, Italy; University of Essex, UK; Technical University of Sofia, Bulgaria; University of California at Berkeley; RIKEN Brain Science Institute, Tokyo; University of Keiserlautern, Germany, and others. He is a member of the Governing Board of the International Neural Network Society (INNS) and the Asia Pacific Neural Network Assembly (APNNA). Kasabov is a co-founder of Pacific Edge Biotechnology Ltd. (www.peblnz.com). More information of Prof. Kasabov can be found on the Web site: http://www.kedri.info.

Igor A. Sidorov was born in August 18, 1961. He received his B.D. and M.D. in Mathematics and Applied Mathematics from Novosibirsk State University, USSR in 1978–1983. He completed his postgraduation in Institute for Numerical Mathematics, Russian Academy of Sciences, Moscow, USSR in 1985–1988 and his Ph.D. in Physics and Mathematics in 1989. From the year 1993, he has been working as a Lecturer at Pushchino State University, Pushchino, Russia. He is the head of the Laboratory of Gene Systematic of Microorganisms, Member of the Academic Council, and Deputy Director (Science) of IBPM RAS, Pushchino, Russian Federation from the year 1998. Sidorov has been awarded the Bellman Prize, for the best paper published in "Mathematical Biosciences" (1992–1993) in the year 1996. He has published 15 research papers in scientific journals.



Dr. Dimiter S. Dimitrov obtained his Ph.D. in chemistry at the University of Sofia in 1976, and his Sc.D. in biology at the Bulgarian Academy of Sciences in 1984, where he was elected as a professor of biophysics in 1988. He was also a visiting scientist at the Cancer Research Laboratory of Carnegie-Mellon University (Rakesh Jain, 1983–84). He joined NIH in 1990 and received tenure in 1993. His current research focuses on protein–protein interactions in virus entry and in cancer cells, and development of human monoclonal antibodies for prevention and treatment of viral diseases and cancer.

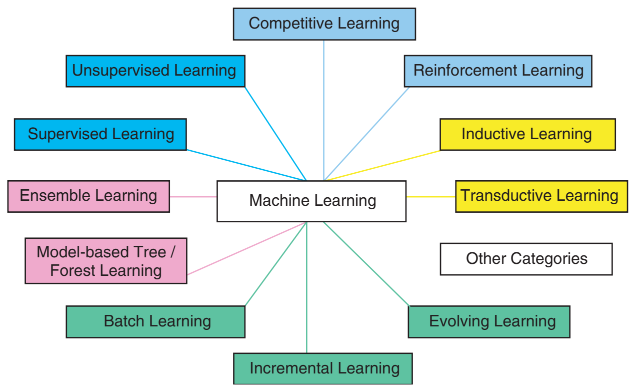


Fig. 1. A diagrammatic representation of the variety of machine learning techniques (ML) used in the CI methods.

intelligence. They are characterised by learning, generalisation, adaptation, pattern recognition, rule extraction, knowledge representation, which are characteristics of the living systems too. There is a large variety of machine learning (ML) techniques used in the above methods as shown graphically in Figure 1.

The methods of CI include:

- Probabilistic learning methods, e.g., Hidden Markov Models;
- Statistical learning methods, e.g., Support Vector Machines (SVM), Bayesian classifiers;
- Case-based reasoning (e.g., k-NN; transductive reasoning);
- Decision trees;
- Rule-based systems (propositional logic dated back to Aristotel) and fuzzy systems (introduced by L. Zadeh[1]);
- Neural networks;
- Evolutionary computation;
- Particle swarm intelligence;
- Artificial Life;
- Quantum computation;
- Hybrid systems (e.g., knowledge-based neural networks; neuro-fuzzy systems; neuro-fuzzy-genetic systems; evolving connectionist systems).

In this section we give a brief description of some of the above methods.

2.1. Probabilistic and Statistical Methods

These methods are based on probability of event estimation and their statistical analysis. Bayesian methods are among the most popular. They are based on the Bayesian probability that represents the conditional probability between two events C and A (Thomas Bayes, 18th century):

$$p(A|C) = \frac{p(A|C)p(A)}{p(C)}$$

Sometimes, using the Bayesian formula involves difficulties, mainly concerning the evaluation of the prior probabilities $p(A)$, $p(C)$, $p(C|A)$.

A very popular statistical technique for discovering patterns in data is clustering. Based on a measured distance between instances (objects, points, vectors) from the problem space, groups of close instances can be defined. These groups are called clusters. They are defined by their cluster centers and the membership of the data points to them. A centre c_i of a cluster C_i is defined as an instance, the mean of the distances to which from each instance in the cluster, is less than its distance to another cluster centre. Let us have a set X of p data items represented in an n-dimensional space. A clustering procedure results in defining k disjoint subsets (clusters), such that every data item (n-dimensional vector) belongs to one only cluster. A cluster membership function M_i is defined for each of the clusters $C_1, C_2, \ldots, C_k$:

$$M_i : X \rightarrow \{0, 1\}$$

$$M_i(x) = \begin{cases} 1, & x \in C_i \\ 0, & x \notin C_i \end{cases}$$

where x is a data instance (vector) from X.

A significant characteristic of clustering is how distance between vectors is measured. The distance between two data points in an n-dimensional geometrical space can be measured in several ways, e.g:

$$\text{Hamming distance:} \quad D_{ab} = \sum |a_i - b_i|$$

$$\text{Euclidean distance:} \quad E_{ab} = \sqrt{\frac{1}{n}\sum (a_i - b_i)^2}$$

A special type of clustering is called fuzzy clustering in which clusters may overlap, so that each of the data instances may belong to each of the clusters to a certain degree.[2] The procedure aims at finding the cluster centres V_i $(i = 1, 2, \ldots, c)$ and the cluster membership functions μ_i which define to what degree each of the n examples belong to the i-th cluster. The number of clusters c is either defined *a priori* (supervised type of clustering), or chosen by the clustering procedure (unsupervised type of clustering). The result of a clustering procedure can be represented as a fuzzy relation $\mu_{i,k}$, such that:

(i) $\sum \mu_{i,k} = 1$, for each $k = 1, 2, \ldots, n$; (the total membership of an instance to all clusters equals 1);
(ii) $\sum \mu_{i,k} > 0$, for each $i = 1, 2, \ldots, c$ (there are no empty clusters).

Probabilistic and statistical methods are used widely in Bioinformatics tools and systems for biological data analysis and modeling.

Regression and discrimination analysis, along with Support Vector Machines (SVM)[3] are popular techniques to build a classifier function. The idea of SVM is to transform original data into higher dimensional feature space via a kernel computation, and to construct a separating hyper

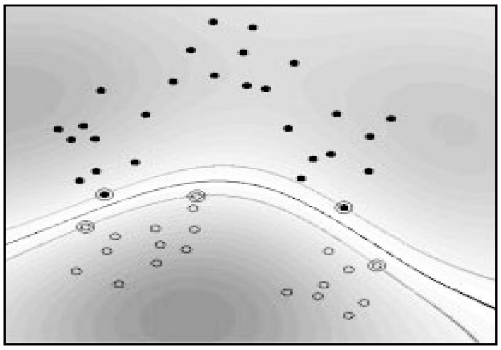


Fig. 2. An example of a separation hyperplane between two class examples in a SVM using a polynomial kernel function.

plane with maximum margin between the class samples (see Fig. 2 as an example). These kernel functions could be polynomial, radial basis, linear, etc.

SVM are widely used for gene expression and protein expression data classification and profiling.

2.1.1. Stochastic Models

Stochastic models deal with the dynamic history of each object of the model. In other word, for each object the next state must be calculated using a set of probabilistic rules. Each rule shows the probability for object to be changed in particular interval of time, and probability to come to each state. So, the change of state in this type of model is probabilistic, not deterministic.

Let us assume that object x in the system has a finite state space with L states (like in a kinetic logic model): $\{X_1, X_2, \dots, X_L\}$. For each time step t_{k+1} there is a transition probability $P(x_{k+1}|x_0, \dots, x_k)$; and a chain $x_0, \dots, x_k$ represents the history of the system. Variables x_k form Markov chain if and only if for any k:

$$P(x_{k+1}|x_0, \dots, x_k) = P(x_{k+1}|x_k)$$

In other words, a future state depends only on the current state. All probability values $P(X_i|X_j)$ that represent the probability for the system to move from the i-th to the j-th state, form a transition matrix.

Suppose that a system can move to state X_i at time a t_k with a transition rate λ_k^i. The probability of the system to move to the i-th state at the time t_k is:

$$p_k^i = \frac{\lambda_k^i}{\lambda}, \quad \lambda = \sum_{i=1}^{L} \lambda_k^i$$

The formula for calculating a next time point depends on the distribution of the moves $t_{k+1} - t_k$, and, for instance, in the case of an exponential process, the next time point is the following: $t_{k+1} = t_k - \ln(r)/\lambda$, where r, is a random value uniformly distributed in (0, 1).

Stochastic models are used for modeling gene regulatory networks (GRN).

2.2. Boolean- and Fuzzy Logic Models

2.2.1. Boolean Models

Consider the set of N objects at time $t_k\{x_1^k, x_2^k, \dots, x_N^k\}$ and each object can be in only two different states: on/off, 1/0, False/True, etc. For simplicity, let us assume:

$$x_i^k \in \{0, 1\}, \quad i = 1, \dots, N$$

The state of the system at a time moment can be described as the states of all objects in this set. The state of a given object at the next time step t_{k+1} can be determined by a Boolean logic function (returning only two values: 0 or 1) that takes as an input the current state of the system:

$$x_i^{k+1} = B_i(x_1^k, x_2^k, \dots, x_N^k)$$

A Boolean function $B = \{B_1, B_2, \dots, B_N\}$ can be represented as a truth table which consists of all possible system states (2^N). This function represents relations between all system's states and can be represented as a diagram. Examples of Boolean functions are given below:

$$A^{k+1} = A^k | B^k$$
$$B^{k+1} = A^k |^\neg B^k$$

where: |, logical OR; and $^\neg$, logical NOT.

Boolean methods are used for gene regulatory network modeling to represent the expression of a gene (1 expressed; or 0 not expressed) and the connection between the genes (1—excitatory, and −1—inhibitory). Boolean models are very limited in terms of representing the "grayness" in biological systems and the "smoothness" in the interaction of its elements.

2.2.2. Fuzzy Logic Models

Fuzzy logic is a logic system that is based on fuzzy relations and fuzzy propositions, the latter being defined on the basis of fuzzy sets.[1] A fuzzy set is a set defined by a membership function to which each domain value can belong to any degree of membership between 0 and 1 and not just 1 (belong) and 0 (does not belong) as in ordinary sets. A variable that can take as values symbolic concepts, such as small, medium, high, each defined by their fuzzy membership function, is called fuzzy variable. Fuzzy propositions are propositions which contain fuzzy variables with their fuzzy values. The truth value of a fuzzy proposition "X is A" is given by the membership function μ_A of the fuzzy value A. Fuzzy relations link two fuzzy sets in a predefined manner. Fuzzy relations makes it possible to represent ambiguous relationships, like: "the expressions of the genes in cluster 2 and cluster 3 are similar," or "model A performed more or less better than model B," or—"the more the gene X is expressed, the higher the risk of cancer."

Fuzzy logic allows for representation of "grayness," "smoothness," inexactness, flexibility, mobility of

Two fuzzy rules of a general type. Each rule has two fuzzy input variables x1 and x2 and one fuzzy output variable.

Rule r_1: IF x1 is Small (DI11) and x2 is Small (DI21) THEN Output is Small (CF1),

Rule r_j: IF x1 is Large (DI1j) and x2 is Large (DI2j) THEN Output is Large (CFj),

where: x1 and x2 are input variables, and Output is an output variable; Small and Large are fuzzy values defined by their respective fuzzy membership functions and DI and CF are degree of importance (membership) and certainty factors respectively. These rules are facilitated in the ANN structure shown in Figure 3 and explained later in the next sub-section.

An exemplar fuzzy rule representing a gene expression profile of a disease:

IF (gene A is highly expressed) **AND**

...

(gene B is lowly expressed) **AND**

(gene C is very highly expressed)

THEN The cancer outcome is likely to be good.

Box 1.

relations, and concepts, which is important when dealing with biological data and concepts. Examples are: "high/low gene expression values;" "strong/week binding between proteins;" "more or less similar structures;" "fast growing culture," and many more.

A fuzzy model is represented usually as a set of fuzzy rules and an inference algorithm. An example of a set of two general fuzzy rules is given in Box 1, where a fuzzy rule for gene expression profiling is shown as a concrete example in the box.

2.3. Artificial Neural Networks

Artificial neural networks (ANN) (connectionist systems) are computational models that mimic vaguely the nervous system in its main functions of adaptive learning and generalization.[4–8] They are universal computational models so that any algorithm or function can be realised as an ANN model. Moreover, ANN can learn functions from data without specifying the type of the function. They are called model-free estimators.

ANNs provide a model of computation that is different from traditional algorithms. Typically, they are not explicitly programmed to perform a given task; rather, they learn to do the task from examples of desired input/output behavior. The networks automatically generalize their processing knowledge into previously unseen situations, and they perform well for the noisy, incomplete or inaccurate input data.

In general view, artificial neural network is the model consisting of interconnected units evolving in time. A connection between units i and j is usually characterized by a weight denoted as w_{ij}. There are three important architectures ANN based on the connectivity:

- Recurrent (contains direct loops from output units (nodes) back to input nodes);
- Feed-forward (contains no direct loops);
- Layered (units organized into layers and connections are between layers).

The behavior of each unit in time can be described by a time-dependent function, or a stochastic process, or a Bayesian formula, etc. So, i-th unit receives total input x from the units connected to it and generates a response based on an activation function. When this function is a threshold function

$$f(x) = \begin{cases} 1, & x > 0 \\ 0, & x \leq 0 \end{cases}$$

the unit is called a threshold gate and can generate only binary decisions.

ANN can implement different ML techniques from Figure 1 and hence—the variety of the ANN architectures. Many of these architectures are known as "black boxes" as they do not facilitate revealing internal relationships between inputs and output variables of the problem in an explicit form. But for the process of knowledge discovery, having a "black box" learning machine is not sufficient. A learning system should also facilitate extracting useful information from data for the sake of a better understanding and learning of new knowledge.

The knowledge-based ANN (KBANN) have been developed for this purpose. They combine the strengths of different AI techniques, e.g., ANN and rule-based systems or fuzzy logic. Rules can be extracted from the KBANN as illustrated in Figure 3.[9]

Evolving connectionist systems (ECOS) have been recently developed to facilitate both adaptive learning in an evolving structure and knowledge discovery.[10] ECOS are modular connectionist-based systems that evolve their structure and functionality in a continuous, self-organised, on-line, adaptive, interactive way from incoming information; they can process both data and knowledge in a supervised and/or unsupervised way. Learning is based on clustering in the input space and on function estimation for this cluster in the output space. Prototype rules can be extracted to represent the clusters and the functions associated with them.

Different types of rules are facilitated by different ECOS architectures, such as evolving fuzzy neural networks (EFuNN)—see Figure 3; dynamic neuro-fuzzy inference systems DENFIS, etc.[10] An ECOS structure grows and "shrinks" in a continuous way from input data streams.

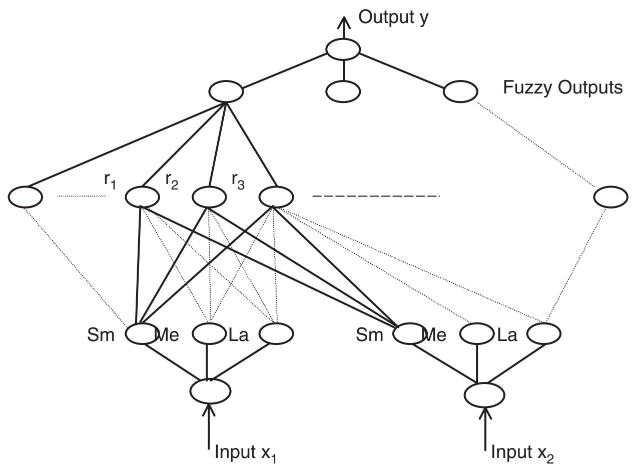


Fig. 3. An artificial neural network model (in this case it is evolving fuzzy neural network EFuNN having two inputs, three fuzzy membership functions and one output).

Feed-forward and feedback connections are both used in the architectures. The ECOS are not limited in number and types of inputs, outputs, nodes, connections. MATLAB codes of EfuNN and DENFIS, as well as some other ECOS techniques, are available from www.kedri.info/. A simple learning algorithm of a simplified version of EFuNN called ECF (Evolving Classifying Function) is given in Box 2.

ECOS have been used for different tasks, including gene expression modeling and profile discovery (see the next section), GRN modeling, protein data analysis, brain data modeling, etc.[10]

2.4. Evolutionary Computation (EC)—Genetic Algorithms (GA)

EC methods are inspired by the Darwinian theory of evolution. These are methods that search in a space of possible solutions for the best solution of a problem defined through an objective function.[11] EC methods have been used for parameter estimation or optimization in many engineering applications. Unlike classical derivative-based (like Newton) optimization methods, EC is more robust against noise and multi-modality in the search space. In addition, EC does not require the derivative information of the objective function and is thus applicable to complex, black box problems. Several techniques have been developed as part of the EC area: genetic algorithms (GA), evolutionary strategies, evolutionary programming, particle swarm optimization, artificial life, etc., the GA being the most popular technique so far.

A genetic algorithm GA is an optimization technique aiming at finding the optimal values of parameters ("genes") for the "best" "individual" according to a

The learning algorithm for the ECF ANN:

1. Enter the current input vector from the data set (stream) and calculate the distances between this vector and all rule nodes already created using Euclidean distance (by default). If there is no node created, create the first one that has the coordinates of the first input vector attached as input connection weights.
2. If all calculated distances between the new input vector and the existing rule nodes are greater than a max-radius parameter Rmax, a new rule node is created. The position of the new rule node is the same as the current vector in the input data space and the radius of its receptive field is set to the min-radius parameter Rmin; the algorithm goes to step 1; otherwise it goes to the next step.
3. If there is a rule node with a distance to the current input vector less then or equal to its radius and its class is the same as the class of the new vector, nothing will be changed; go to step 1; otherwise:
4. If there is a rule node with a distance to the input vector less than or equal to its radius and its class is different from those of the input vector, its influence field should be reduced. The radius of the new field is set to the larger value from the two numbers: distance minus the min-radius; min-radius. New node is created as in 2 to represent the new data vector.
5. If there is a rule node with a distance to the input vector less than or equal to the max-radius, and its class is the same as of the input vector's, enlarge the influence field by taking the distance as a new radius if only such enlarged field does not cover any other rule nodes which belong to a different class; otherwise, create a new rule node in the same way as in step 2, and go to step 1.

Recall procedure (classification of a new input vector) in a trained ECF:

1. Enter the new input vector in the ECF trained system. If the new input vector lies within the field of one or more rule nodes associated with one class, the vector is classified in this class.
2. If the input vector lies within the fields of two or more rule nodes associated with different classes, the vector will belong to the class corresponding to the closest rule node.
3. If the input vector does not lie within any field, then take m highest activated by the new vector rule nodes, and calculate the average distances from the vector to the nodes with the same class; the vector will belong to the class corresponding to the smallest average distance.

Box 2.

A GA algorithm:

GA1. Create a population of N individuals, each individual being represented as a "*chromosome*" consisting of values (alleles) of parameters called "genes."
GA2. *Evaluate the fitness* of each individual towards a pre-defined objective function. If an individual achieves a desired fitness score, or alternatively—the time for running the procedure is over, the GA algorithm STOPS.
GA3. Otherwise, select a subset of "best" individuals using a pre-defined *selection criteria* (e.g. top ranked, roulette-wheel, keep the best individuals through generations, etc.)
GA4. Crossover the selected individuals using a crossover ("mating") technique to create a new generation of a population of individuals.
GA5. Apply *mutation* using a mutation technique. Go to GA2.

Box 3.

pre-defined objective function (fitness function). A GA includes the steps shown in Box 3.

GA is a heuristic and non-deterministic algorithm. It can give a close to optimal solution depending on the time of execution. For a large number of parameters ("genes in the chromosome") it is much faster than an exhaustive search and much more efficient.

Representing real genes, or other biological variables (proteins, binding strengths, connection weights, etc.,) as GA "genes," is a natural way to solve difficult optimization tasks in CB. For this reason GAs are used for several tasks in this paper and also in the proposed in Section 3 method for feature (genes, this time—real ones) and model parameter optimization illustrated on a gene expression classification model.

3. PROBLEMS, METHODS, AND PERSPECTIVES IN COMPUTATIONAL BIOLOGY AND BIOINFORMATICS: A BRIEF OVERVIEW

3.1. General Overview

A major goal of Computational Biology (CB) and Bioinformatics (BI) is to discover knowledge or enhance knowledge discovery for biological systems through computation. Computational approaches provide an underpinning for the integration of broad disciplines for development of a quantitative systems approach to understanding the mechanisms determining the life of the cell and organism. Another aspect of the integration of computation and biology is that biological systems can be viewed as special computing devices. This view emerges from considerations of how information is stored in and retrieved from the genes. Genes can only specify the properties of the proteins they code for, and any integrative properties of the system must be "computed" by their interactions. This provides a framework for analysis by simulation and sets practical bounds on what can be achieved by reductionist models.

Recent advances in many areas of biology, especially in genomics, are heavily rooted in engineering technology, from the capillary electrophoresis units used in large DNA sequencing projects, to the photolithography and robotics technology used in chip manufacture, to the confocal imaging systems used to read those chips, to the beam and detector technology driving high-throughput mass spectroscopy. Further advances in materials science and nanotechnology promise to improve the sensitivity and cost of these technologies greatly in the near future. Current research makes it possible to look at biological phenomena on a scale not previously possible: all genes in a genome, all transcripts in a cell, and all metabolic processes in a tissue.

One core aspect of research in computational biology focuses on data integration: how to integrate and optimally query and analyse data from genomic DNA sequence, spatial and temporal patterns of mRNA expression, protein structure, immunological reactivity, clinical outcomes, publication records, and other sources. A second focus involves pattern recognition algorithms for such areas as nucleic acid or protein sequence assembly, sequence alignment for similarity comparisons or phylogeny reconstruction, motif recognition in linear sequences or higher-order structure, and common patterns of gene expression and proteins. Some of these problems along with possible solutions are discussed in the rest of the paper.

3.2. Computational Genomics

DNA (deoxyribonucleic acid) is a nucleic acid polymer consisting of individual units termed nucleotides. Each nucleotide consists of one of four distinct nucleosides (deoxypentose sugar plus one of four bases (Adenine (A), Guanine (G), Cytosine (C), and Thymine (T))) and a phosphate group. Thymine is replaced by Uracil (U) in RNA (ribonucleic acid). With respect to similarity in structure, nucleosides are divided in two classes: pyrimidines and purines. Nucleosides A, T, G, and C are capable of being linked together to form a long chain. The bases along the polymer can interact with complementary bases in the other strand: adenine is capable of forming hydrogen bonds with thymine (A:T) and cytosine can pair with guanine (C:G). A nucleoside is one of the four DNA bases attached covalently to the sugar. The sugar in deoxynucleosides is 2′-deoxyribose and ribose in ribonucleosides. The four different nucleosides of DNA are deoxyadenosine (dA), deoxyguanosine (dG), deoxycytosine (dC), and deoxythymidine (dT). A nucleotide is a nucleoside with one or more phosphate groups covalently attached to the 3′- and/or 5′-hydroxyl group(s). The DNA backbone is a polymer with an alternating sugar–phosphate sequence. DNA is a normally double stranded macromolecule with two polynucleotide chains (the double helical nature of DNA was discovered in 1953.[12]). These chains are noncovalently held together by weak intermolecular forces and form a DNA molecule. Two DNA strands form a helical spiral, winding around a helix axis in a right-handed spiral with two polynucleotide chains running in opposite directions. The sugar–phosphate backbones wind around the helix axis. The bases of the individual nucleotides are on the inside of the helix. For DNA duplexes, the right handed double helix has 10 pairs per complete turn. Within the DNA double helix, the adenine:thymine base pair has two hydrogen bonds compared to three in the guanine:cytosine pair. The two base pairs are required to be identical in dimensions by the Watson-Crick model. High resolution X-ray crystallographic analysis of the ribodinucleoside monophosphate duplexes (G:C and A:U) showed that the distance between the glycosidic carbon atoms in the base pairs are close (10.67 Å and 10.48 Å, respectively).

RNA molecules are polynucleotides containing ribose sugars connected by phosphodiester linkages. Although RNA is generally single-stranded, double-stranded RNA molecular can be formed where uracil participates in U:A pair. Single-stranded RNA have a tendency to fold back on themselves to form double-stranded structures like stacked double helix for the regions with paired bases and different loops (bulge, hairpin, internal, and multibranch) for unpaired ones. These elements form the RNA secondary structure.

Prediction of RNA secondary structure requires intensive computational resources. Usually RNA resultant structure corresponds to the local minima of the free energy and overall free energy of the molecular folded is the sum of the energies of the stacked base pairs and loops. However molecular environment and folding pathway, which can have significant impact on this structure, should be accounted.

In Structurelab[13] dynamic programming algorithm and a GA were used for determination of the folding of the RNA molecules. This system allows researchers to pursue interactively and methodically a multiperspective analysis of RNA structure (multiple and individual). It utilizes various software modules and hardware complexes.[14] Secondary structure representation of RNA molecular structures is based on LISP's nested list notations, for instance (N(H)(H)(BH)(H)(H)(H)(BBBIH)), where symbols are: H, hairpin loop, B, bulge loop, I, internal loop, M, multibranch loop.

Other packages that can be used for secondary structure prediction and presentation are available from the following WWW sites: http://www.bioinfo.rpi.edu/∼zukerm/rna/, http://bioweb.pasteur.fr/seqanal/interfaces/mfold-simple.html, http://biotools.idtdna.com/mfold/, http://www.bioinfo.rpi.edu/applications/mfold/old/rna/form1.cgi; http://rna.tbi.univie.ac.at/cgi-bin/RNAfold.cgi; http://www.inra.fr/bia/T/essa/Doc/essa_home.html, http://rrna.uia.ac.be/card.html.

3.3. Searching for Motifs in Sequences

To find a motif is to find a pattern consensus between a specified new sequence, and a given existing one. Let us consider a sequence as a vector of symbols:

$$X = (x_1, x_2, \dots, x_L)$$

where: L, sequence length and all symbols x_i belong to a finite set of symbols (or alphabet):

$$x_i \in A = \{a_1, \dots, a_K\}, \quad i = 1, \dots, L$$

For DNA sequences the alphabet is simply set of four letters:

$$A = \{A, T, G, C\}$$

Searching for functional motif can be considered as a comparing two sequences; one is the target sequence (S) and another one is the motif (M):

$$S = (s_1, s_2, \dots, s_{L_S}), \quad M = (m_1, m_2, \dots, m_{L_M})$$

having length L_S and L_M, respectively.

Let now assume that function $w_{ij} = w(a_i, a_j)$ expresses weight of combination of two symbols in comparing sequences. The simplest equation for this function is:

$$w(a_i, a_j) = \begin{cases} 1, & a_i = a_j \\ 0, & a_i \neq a_j \end{cases}$$

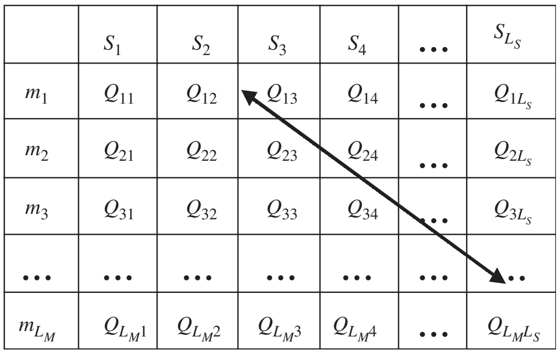
In this case, position j of the motif in target sequence can be found as a value giving maximum for the following score:

$$Q_j = \sum_{i=1}^{L_M} w(m_i, s_{i+j-1}), \quad j = 1, \dots, L_S - L_M + 1$$

Motif can be also represented as a probability (frequency or weight) $p_i(a_j)$ to find j-th symbol a_j from alphabet A in position i of the motif. Using the values of matrix $\|p_i(a_j)\|$ (position weight matrix, PWM), the score value for each position can be calculated as:

$$Q_j = \sum_{i=1}^{L_M} p_i(s_{i+j-1}), \quad j = 1, \dots, L_S - L_M + 1$$

If for i-th position in the motif $p_i(m_i) = 1$ and equals 0 for all other symbols this approach is equal to the simplest case of the weight function. Graphically both methods can be represented as shown in the Figure 4. The matrix shown



	S_1	S_2	S_3	S_4	…	S_{L_S}
m_1	Q_{11}	Q_{12}	Q_{13}	Q_{14}	…	Q_{1L_S}
m_2	Q_{21}	Q_{22}	Q_{23}	Q_{24}	…	Q_{2L_S}
m_3	Q_{31}	Q_{32}	Q_{33}	Q_{34}	…	Q_{3L_S}
…	…	…	…	…	…	…
m_{L_M}	Q_{L_M1}	Q_{L_M2}	Q_{L_M3}	Q_{L_M4}	…	$Q_{L_ML_S}$

Fig. 4. Example for a motif search in a sequence.

in this picture is dot plot, i.e., visual representation for the similarity between two sequences. Each axis represents one of the two sequences to be compared and each cell represents weight or probability function values. To calculate Q_j for position j in target sequence one should summarize values on the diagonal starting from the cell $(1, j)$. For the case when weight function has a simplest representation (=1 for the equal symbols and 0 for different ones) and values of this function are represented as empty and filled cell for 0 and 1, respectively, the whole diagonal will represent sequences sharing similarity. For sequences which share only patches of similarity diagonal stretches will be shown.

The method of dotmatrix analysis was first described in Ref. [15]. It can be also useful for finding inverted repeats and self-complimentary repeats. The use of enhanced dot plot for nucleic and protein sequences was described in Ref. [16]. Additional description of the method can be found in Ref. [17].

A package for detection of patterns and structural motif in nucleotide sequences (PatSearch) is described in Ref. [18]. It allows scanning for specific combinations of oligonucleotide consensus sequences with defined order, orientation and spacing, and allowing also mismatches and mispairing below a user fixed threshold (available at http://bighost.area.ba.cnr.it/BIG/PatSearch). The possible pattern units for this package are: string, palindrome, hairpin loop, position weight matrix, repeat. It uses also logical patterns as "either/or" and length constraints for specific combination of pattern units.

The TRANSFAC database (available at http://www.gene-regulation.com.) on eukaryotic transcriptional regulation, comprising data on transcriptional factors, their target genes and regulatory binding sites[19] and tools for matrix-based search of transcription factors binding sites (MATCH). The algorithm of MATCH uses two values to score hints: the matrix similarity score and the core similarity score which is close to the MatInspector algorithm.[20]

3.4. Sequence Alignment

There are two main type of sequence alignment: pairwise (comparing two sequences) and multiple sequence alignment (comparing more than two sequences). Multiple sequence alignment is the procedure of comparing sequences by searching for the similarity in the subsets that are in the same order in the sequences. Each subset can consists of the one or more character of the sequence and gap(s) between them.

Comparison of two or more sequences has strong biological rationales. One of them is the fact that gene sequences may have evolved from common ancestral sequences and thus the changes in sequence (mutation, insertion, and deletion) can show us the evolution course of the particular molecule. Another reason is indicating the regions of common origin which may in turn coincide with regions of similar structure or similar function. Results of alignment can be used as a starting point for solving various tasks (predicting de novo the secondary structure of proteins and other knowledge-based structure predictions; resolving phylogenetic issues; interpreting data from the human genome).

Let us have n sequences $S_1, S_2, \ldots, S_n$, and each sequence be represented as a vector:

$$S_i = (s_1^i, s_2^i, \ldots, s_{L_i}^i), \quad i = 1, \ldots, n$$

where: N_i is the length of the i-th sequence, and

$$s_j^i \in A = \{a_1, \ldots, a_K\}, \quad \text{for all } i \text{ and } j$$

Let now assume that each sequence can be represented with gaps (insertions/deletion) so:

$$S_i = (g_0^i, s_1^i, g_1^i, s_2^i, g_2^i, \ldots, g_{L_i-1}^i, s_{L_i}^i, g_{L_i}^i)$$

where g_j^i, $j = 0, \ldots, L_i$ are the gaps inserted in the i-th sequence at j-th place. Alignment of n sequences can be represented as a matrix $R = \|r_{ij}\|$, with the following properties:

- $r_{ij} \in A \cup \{\text{gap}\}$, so gap is included in alphabet.
- each row matrix represents i-th sequence with gaps: $r_i = S_i$.
- each column can not consist only of gaps.

Score function for multiple alignments depends on the weight function ($w_{ij} = w(a_i, a_j)$, scoring matrix) and so-called gap-penalty function. The latter describes the decrease of score for gaps of given length and consists of the constant term describing penalty for opening the gap (a) and penalty for each element in gap (b). The usual formula for penalty of gap having length is (so-called affine gap penalty):

$$Q(g) = a + bg$$

and one of its extension:

$$Q(g) = \begin{cases} a + b(g - q), & g > q \\ a, & g \le q \end{cases}$$

where q means that gap penalty for each element will be added only when gap size is greater than q.

It is obvious that gap penalty function have to be appropriate to the weight function to obtain a reasonable alignment. If the gap penalty function is high enough with respect to scoring matrix values final alignment will never have gaps. On the other hand, too small values of the gap penalty function will lead to the alignment with gaps occurring everywhere.

Two alignments can be compared using the same score function. A key element in evaluating the quality of a sequence alignment is the score matrix (or substitution score matrix) $w_{ij} = w(a_i, a_j)$, which assigns a score for aligning any possible pair of sequence elements. The theory of amino acid substitution matrices is described in Ref. [21], and applied to DNA sequence comparison in Ref. [22].

Basic local alignment search tool (BLAST) for rapid sequence comparison was developed in 1990.[23] It directly approximates alignments that optimize a measure of local similarity, the maximal segment pair (MSP) score. The basic algorithm is simple and robust and it was applied in straight-forward DNA and protein sequence database searches, motif searches, gene identification searches, and in the analysis of multiple regions of similarity in long DNA sequences. A new generation of this algorithm for searching databases (gapped BLAST and PSI-BLAST) is described in Ref. [24].

3.5. Computational Proteomics

Proteins are linear polymers of amino acids linked by a peptide bond. Amino acids are small molecules consisting of amino group (NH_2), a carboxyl group (COOH), hydrogen atom attached to the central carbon (α) and side chain (or R group) attached to the central carbon. There are 20 standard amino acids, which can be grouped into classes based on the chemical properties conferred by their side chains. Amino acid can be charged (+/−), hydrophobic/hydrophilic, polar/nonpolar capable of H-bonding—allowing for weak interactions. Amino acids can form peptide bonds with each other through reaction of the carboxyl and amino groups.

Understanding the structures, interactions, and functions of all of a cell's or organism's proteins has been given a disciplinary title of its own: *proteomics.* The ultimate goal of proteomics is to characterize the information flow through protein networks.

The word proteome, coined in 1994 as a linguistic equivalent to the concept of genome, indicates proteins expressed by a genome. This term was coined by Marc Wilkins and colleagues and appeared for the first time in 1995.[25] The term proteome is used to describe the complete set of proteins that is expressed, and modified following expression, by the entire genome in the lifetime of a cell. It can be used also as a description of proteins expressed by a cell at any stage. The generation of messenger RNA expression profiles is referred to as transcriptomics, as these are based around the process of transcription. And the mRNAs transcribed from a cell's genome is the transcriptome.

The *primary structure* of the protein (the liner protein sequence) determines the ultimate three-dimensional structure of the protein. The *secondary structure* represents the local folding of peptide that results in distinctive structures shared by many proteins, including alpha (α) helices and beta (β) pleated sheets. These structures were predicted theoretically prior to the experimental determination of protein structure and they are the only regular secondary structural elements present in proteins (there are also irregular structural elements: loop and coil). Helix is created by a curving of the polypeptide backbone and sheet is formed by hydrogen bonds between adjacent polypeptide chains rather than within a single chain. There are two configurations for both elements: rightward/leftward for helix and parallel/antiparallel for sheet.

The tertiary structure is the global 3D structure of the polypeptide chain. At this level of structure the side chains play a major role in creating of the final structure. Protein folding is a process of forming final three-dimensional tertiary structure. It is interesting to note that random polypeptide sequences almost never fold into an ordered structure, so, protein sequences were selected by the evolution to achieve reproducible stable structure.[26] *The quaternary structure* represents the interaction of multiple subunits of a protein. Many proteins are formed from more than one polypeptide chain, i.e., exist as a noncovalent association of two or more identical or different polypeptides folded independently. The quaternary structure describes the way in which the different subunits are packed together to form the overall structure of the protein. For example, the human hemoglobin molecule is made of four subunits.

Predicting protein structures is one of the tasks in the computational structural biology. Its main goal is in general to predict the structure and the structural basis of the function of a biologically related molecule. Of all major classes of biomolecules including proteins, DNA, RNA, carbohydrates, and small molecules with biological activity, protein structures have been mostly studied computationally because of their importance and variety of structures known.

Protein *secondary structure prediction* can be performed by different packages, among them:

— NNPREDICT (http://www.cmpharm.ucsf.edu/~nomi/nnpredict.html), that uses a two-layer, feed-forward ANN (see for details Refs. [27, 28]).
— PHDsec (http://cubic.bioc.columbia.edu/predictprotein) predicts secondary structure from multiple sequence alignments. Secondary structure is predicted by an ANN rating for the three states helix, strand, and loop at an expected average accuracy >72%.[29–31]

— PROFsec (http://cubic.bioc.columbia.edu/predictprotein)—an improved version of PHDsec (a profile-based ANN prediction of protein secondary structure).
— JPRED, http://jura.ebi.ac.uk:8888/, a consensus method for protein secondary structure prediction.[32]

Qian and Sejnowski[33] investigated the use of multilayer perceptrons ANN for the task of predicting secondary structure based on available labeled data. In Ref. [10] an Evolving Fuzzy Neural Network (EFuNN) is trained on data from Ref. [33] to predict the shape of an arbitrary new protein segment: a window of 13 aminoacids was used; there were 273 inputs and 3 outputs and 18,000 examples for training. The block diagram of the EFuNN model is given in Figure 3 (from Ref. [10]).

Prediction of three-dimensional structure of proteins can be achieved by homology modeling based on the similarity of primary sequences of the protein being analyzed to a protein of experimentally determined structure (this method assumes that significant identity between the two sequences exists). Algorithms for homology modeling can be found in the following servers:

— SWISS-MODEL (http://www.expasy.ch/swissmod/SWISS-MODEL.html), an Automated Protein Modeling Server running at the GlaxoWellcome Experimental Research in Geneva, Switzerland (see for details http://www.expasy.ch/swissmod/SWISS-MODEL.html and Refs. [34, 35]).
— CPHmodels (http://www.cbs.dtu.dk/services/CPHmodels/), Centre for Biological Sequence Analysis; The Technical University of Denmark; Denmark. Methods and databases developed to predict protein structures: Sowhat, neural network based method to predict contacts between C-alpha atoms from the amino acid sequence; RedHom, tool to find a subset with low sequence similarity in a database (see Ref. [36]).
— 3D-JIGSAW Comparative Modeling Server (http://www.bmm.icnet.uk/~3djigsaw/); BioMolecular Modeling Group; Imperial Cancer Research Fund; London, UK. Automated system to build three-dimensional models for proteins based on homologues of known structure (Refs. [37–39]).
— SDSC1-SDSC Structure Homology Modeling Server (http://cl.sdsc.edu/hm.html) and Databases and Tools for 3-D Protein Structure Comparison and Alignment (http://cl.sdsc.edu/ce.html); San Diego Supercomputing Centre; San Diego, CA, USA (Refs. [40, 41]).

Detailed review of the Bioinformatic tools for proteome profiling can be found in Refs. [42–44].

Development of software for 2-D gel protein image analysis began about 35 years ago[45–47] with further improvement that were made in the late 1980s.[48–52] There are many commercially available packages now, among them: DeCyder 2D Analysis, ImageMaster 2D Elite, http://www1.amershambiosciences.com/; Delta2D, www.decodon.com; GELLAB II+, www.scanalytics.com, http://www-lecb.ncifcrf.gov/lemkin/gellab.html; GeneData Impressionist system, www.genedata.com; ImagepIQ, www.proteomesystems.com; Melanie 3, www.genebio.com; ProteinMine, www.scimagix.com; TotalLab, www.totallab.com. Some of the 2-D gel image analysis packages can interact with automatic robotic systems.

Comparative (homology) modeling is a computational biology method that can provide protein structure prediction with a root-mean-square (rms) error lower than 2 Å. Computational methods for protein structure prediction based on related proteins of known structures have been developed more than three decades ago.[53] Later Greer outlined a basic protocol that is still followed today.[54, 55] Most homology modeling methods consist of four sequential steps.[56] The first step is to identify the proteins with known 3D structures that are related to the target sequence. The second step is to align them with the target sequence and to pick those known structures that will be used as templates. Any corrections in the alignment are made at this stage. The third step is to build the model for the target sequence given its alignment with the template structures. In the fourth step, the model is evaluated using a variety of criteria. If necessary, the alignment and model building are repeated until a satisfactory model is obtained. The main difference between the different comparative modeling methods is how the 3D model is calculated from a given alignment. Because of the importance of step 3 sometimes it is divided into four stages:[57] backbone generation, loop modeling, side-chain modeling, and model optimization.

3.6. Computational Cell Biology

Computational cell biology is an emerging discipline that responds to the need for computational methods to analyze and organize the abundance of experimental data on the structure and function of the cell. Historically (Ref. [58]), mathematical biology has had a limited success turning in time into somewhat abstract discipline. Several examples of early success of mathematical modeling in biology was demonstrated in the following works: Lotka-Volterra predator-prey model in ecology,[59, 60] Hodgkin-Huxley's model of nerve conduction,[61] Manfred Eigen's theory of molecular evolution,[62] Gierer-Meinhardt's theory of biological pattern formation.[63] Extensions of these models were considered later by many biologists as a mathematical refinement with limited practical utility (with some exceptions for the mathematical models in neurobiology and cardiology). Biological systems are complex and open, various factors can change the behavior of the system, so even simple computational modeling requires the continuous interaction between model building and experimental verification.

Complexity of biological objects can be defined as "large number of functionally diverse, and frequently multifunctional, sets of elements which interact selectively and nonlinearly to produce coherent rather then complex behavior."[64] Biological events occurring at various levels (such as organism, tissue, cell, and molecular) and complexity of the system being modeled, lead to the need for integration of different models. For instance, for modeling transduction of activation signal into cell one may need to include gene regulatory network, models of proteins pathways, models of membrane and diffusion of molecules and ions into cell, etc. Some of these models may be available for the investigator but they are most likely different in format, programming languages and computing platforms, so one may need to develop the tools unification of the models and of communication between them. Currently, there are two ongoing projects for introducing standards in the model communication: System Biology Markup Language (www.cds.caltech.edu/erato/sbml/docs) and CellML (www.cellml.org).

Next challenge related to modeling complex biological systems is modeling adaptation, i.e., how a model should change with respect to new experimental data. For a model consisting of hundreds of equations and parameters adaptation to new experimental data is not trivial at all. A possible way to solve this problem is to identify the semi-autonomous functional units in the model with known parameters and system behavior. So, building a complex model or estimating model parameters can be accomplished in a step-wise or modular based manner. The issue of model adaptation and parameter and feature optimization with the use of GA is illustrated in the next sub-section.

3.7. Microarray Gene Expression Data Analysis and Disease Profiling

The recent advent of cDNA microarray and gene-chip technologies means that it is now possible to simultaneously interrogate thousands of genes. The potential applications of this technology are numerous and include identifying markers for classification, diagnosis, disease outcome prediction, therapeutic responsiveness, and target identification. Microarray analysis might not identify unique markers (e.g., a single gene) of clinical utility for a disease because of the heterogeneity of the disease, but a prediction of the biological state of disease is likely to be more sensitive by identifying clusters of gene expression (profiles).[65]

Each point (pixel, cell) in a microarray matrix represents the level of expression of a single gene. Five principal steps in the microarray technology are shown in Figure 5. They are: tissue collection; RNA extraction; microarray gene expression evaluation; scanning and image processing; data analysis.

One of the contemporary directions while searching for efficient drugs for many terminal illnesses, such as cancer

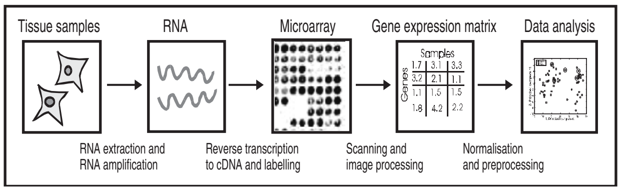

Tissue samples
RNA
Microarray
Gene expression matrix
Data analysis
Samples
Genes
1.7 3.1 3.3
3.2 2.1 1.1
1.1 1.5 1.5
1.8 4.2 2.2
RNA extraction and RNA amplification
Reverse transcription to cDNA and labelling
Scanning and image processing
Normalisation and preprocessing

Fig. 5. The pathway of the microarray technology.

or HIV, is the creation of gene profiles of these diseases and subsequently finding targets for treatment through gene expression regulation. A gene profile is a pattern of expression of a number of genes that is typical for all, or for some of the known samples of a particular disease. An example of a disease profile is shown in Box 1.

Having such profiles for a particular disease makes it possible to set an early diagnostic test, so a sample can be taken from a patient, the data related to the sample processed, and a profile related to the sample—obtained. This profile can be matched against existing gene profiles. Based on similarity, it can be predicted with certain probability if the patient is in an early phase of a disease or he/she is at risk of developing the disease in the future with certain probability.

ANN have been used to create classification systems based on gene expression data. In Ref. [66] a multilayer perceptron ANN was used to achieve a classification of 93% of Ewings sarcomas, 96% of rhabdomyosarcomas, and 100% of neuroblastomas. From within a set of 6567 genes, 96 genes were used as variables in the classification system. Whether these results would be different using different classification methods needs further exploration.

3.7.1. *Feature and Model Optimization for Gene Expression Data*

An usual way of processing gene (and also protein) expression data S of genes G consists of the following steps:

```
(1) FOR $i := 1$ to $N$ (in particular, it could be leave-one-out method, where $N$ is the number of available samples) DO
      (a) select a sub-set Sv, $i$ from S of data for validation;
      (b) for the rest of the data St, $i$ select the most discriminative subset of genes Gi; Card(Gi) $\ll$ Card(G);
      (c) create a model Mi based on the data St, i, and the gene set Gi as input variables;
      (d) validate Mi on the test set Sv, I and calculate the error Ei.
    END (FOR)
(2) Calculate the average error from Ei.
(3) Define a set Gm of the most frequently selected genes in all $N$ iterations.
(3) Create a final model M based on the whole data S and the set of genes Gm.
```

GA optimisation of a gene set and model parameters (see a general description of GA in Box 3)

```
(1) Starting with an initial gene set Gm and a model M, create a population of K chromosomes (models), each having
    different gene subsets from Gm and slightly different parameter values from the parameter values of the model M. The
    chromosome contains a binary part, where a gene is present (1) or not present (0) in a model, and a part of continuous
    values—the parameters of the model. If, for example, the model is ECF (see Box 2), the parameters are: Rmax, Rmin,
    Number of fuzzy membership functions, Number of iterations of training the ECF model.
(2) FOR $J = 1$ to $P$ generations DO
    (a) Select randomly from the data set S a subset Stst for testing and the rest Str for training.
    (b) Train all K models on Str and test them on Stst.
    (c) Select (see Box 3) the best models (e.g., maximum accuracy).
    (d) Apply cross validation and mutation (see Box 3) to the chromosomes to create the next generation of K new
        models.
    END (FOR)
(3) Select the best model (the model with the best accuracy) that has an optimized gene set and optimized model parameter
    values.
```

Box 4.

After the above steps, a model M is created using the most frequently appearing genes in all *N* iterations—a gene set Gm. The gene set Gm though may not be optimal in terms of representing a minimum set of genes that through their interaction "cover" all clusters of the problem space. The parameters of the model M derived above were not optimized and may not be optimal either. Box 4 represents a GA optimization algorithm for the optimization of both the gene set and the model parameter values as an additional procedure to the procedure above.

Example is given in Figure 6a (as a screen dump of the procedure used in the software environment SIFTWARE). A well known DLBCL cancer outcome prognosis data set is used (Shipp et al., 2002). As a starting point for the gene set optimization, the set of 11 genes selected as the most frequent in this publication, are used for the ECF classification model with default parameter values (see Box 2). The cross validation accuracy obtained was 88%. When the algorithm from Box 4 is applied, the accuracy after 5 fold cross validation is 91% and the selected optimal variables are: 1, 2, 3, 4, 5, 7, 9, 10, 11, where variable 1 is the IPI clinical prognostic index and the rest of the variables from 2 to 12 correspond to the genes from 1 to 11 in (Shipp et al., 2002). The procedure in Box 4 results in a better accuracy model and in a smaller gene set (eight) that any of the published results. This case study also illustrates one of the problems of integrating different sources of information, in this case—gene and clinical variables, in an optimized model.

Using the best ECF model evolved as shown in Figure 6a, a set of fuzzy rules can be extracted from it, each representing a cluster of data and interpreted as a profile of the cluster for the respective class. This is shown in Figure 6b, where for class 1 (good outcome) the clusters are 12 and for class 2 (fatal outcome) the clusters are 11. Red color represents a high value of a variable and green color represents a low value.

3.8. Computational Systems Biology

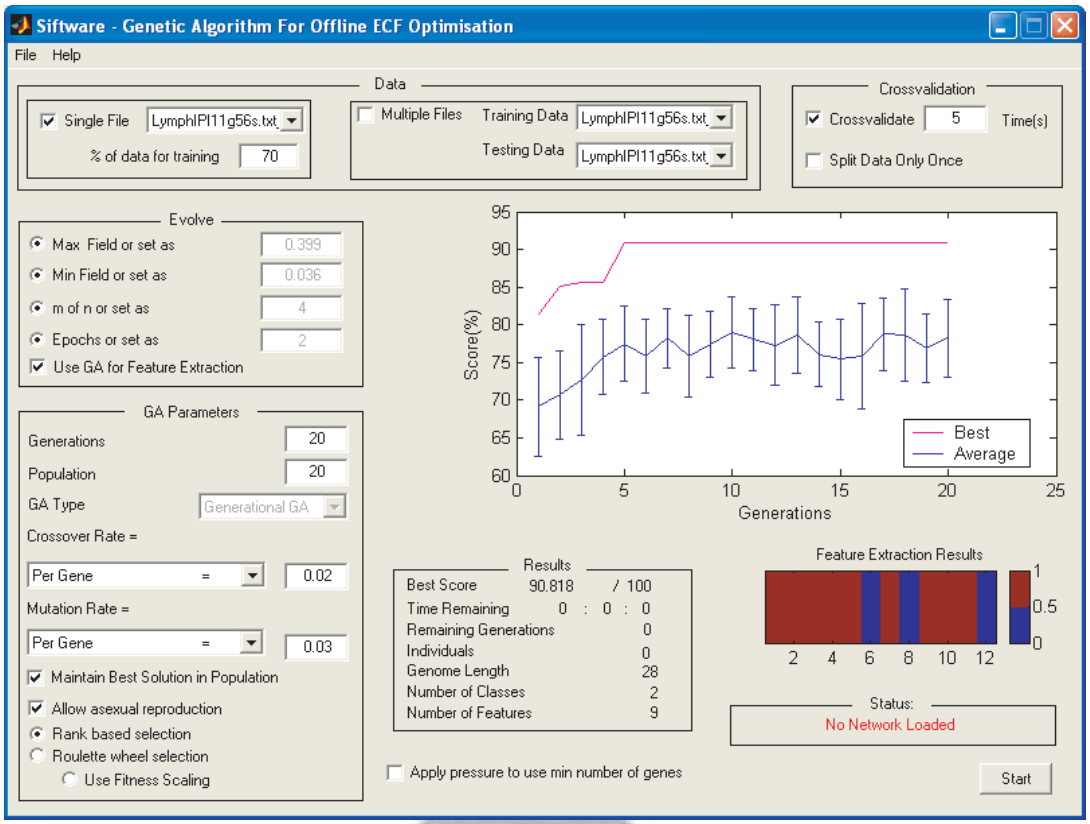
The aim of computational systems biology is to understand complex biological objects in their entirety, i.e., at a system level. It involves the integration of different approaches and tools: computer modeling, large-scale data analysis, and biological experimentation. One of the major challenges of the systems biology is the identification of the logic and dynamics of gene-regulatory and biochemical networks. The most feasible application of systems biology is to create a detailed model of a cell regulation to provide system-level insights into mechanism-based drug discovery.[67–69]

System-level understanding is a recurrent theme in biology and has a long history.[70–72] The term "system-level understanding" was described in Ref. [73] as the shift of focus in understanding a system's structure and dynamics in whole rather than the particular objects and their interactions. System-level understanding of a biological system can be derived from insight into four key properties:[64]

System structures—these include the gene regulatory network (GRN) and biochemical pathways. They can also include the mechanisms of modulation the physical properties of intracellular and multicellular structures by interactions.

System dynamics. System behavior over time under various conditions can be understood by identifying essential mechanisms underlying specific behaviors and through various approaches depending on the systems nature: metabolic analysis (finding a basis of elementary flux modes that describe the dominant reaction pathways within the network), sensitivity analysis (the study of how the variation in the output of a model can be

(a)



Siftware - Genetic Algorithm For Offline ECF Optimisation

File Help

Data

[✓] Single File LymphIPI11g56s.txt

% of data for training 70

[ ] Multiple Files Training Data LymphIPI11g56s.txt

Testing Data LymphIPI11g56s.txt

Crossvalidation

[✓] Crossvalidate 5 Time(s)

[ ] Split Data Only Once

Evolve

[✓] Max Field or set as 0.399

[✓] Min Field or set as 0.036

[✓] m of n or set as 4

[✓] Epochs or set as 2

[✓] Use GA for Feature Extraction

GA Parameters

Generations 20

Population 20

GA Type Generational GA

Crossover Rate =

Per Gene = 0.02

Mutation Rate =

Per Gene = 0.03

[✓] Maintain Best Solution in Population

[✓] Allow asexual reproduction

[✓] Rank based selection

[ ] Roulette wheel selection

[ ] Use Fitness Scaling

Results

Best Score	90.818 / 100
Time Remaining	0 : 0 : 0
Remaining Generations	0
Individuals	0
Genome Length	28
Number of Classes	2
Number of Features	9

Feature Extraction Results

Status:

No Network Loaded

[ ] Apply pressure to use min number of genes

Start

(b)

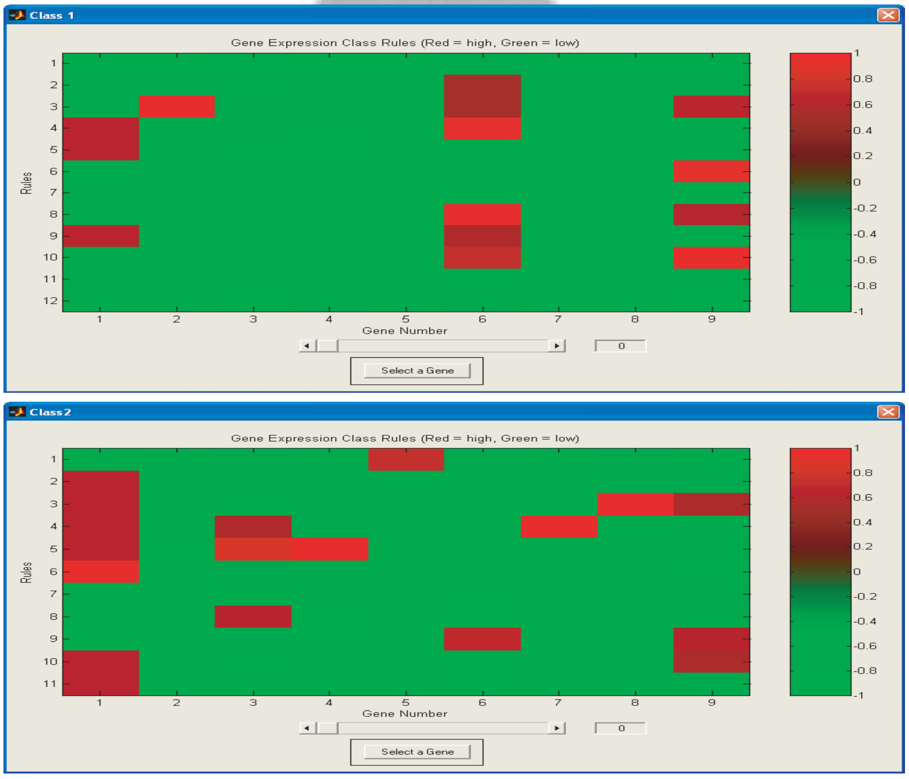


Fig. 6. Gene set and model parameter optimization for the DLBCL cancer outcome prognosis data with use of SIFTWARE (www.peblnz.com): (a) the screen dump of the results from the optimisation procedure: (b) a fuzzy rule (a profile) for each cluster of the DLBCL cancer outcome prognosis data is derived from a trained ECF (see Box 2) using the optimized parameter values and inputs for the two classes—class 1 (a good outcome) and class 2 (a fatal outcome).

apportioned, qualitatively or quantitatively, to different sources of variation), dynamic analysis methods such as phase portrait (geometry of the trajectories of the system in state space) and bifurcation analysis (bifurcation analysis traces time-varying change(s) in the state of the system in a multidimensional space where each dimension represents a particular system parameter (concentration of the biochemical factor involved, rate of reactions/interactions, etc.). As parameters varied, changes may occur in the qualitative structure of the solutions for certain parameter values. These changes are called bifurcations and the parameter values are called bifurcation values).

The control method. Mechanisms that systematically control the state of the cell can be modulated to change system behavior and optimize potential therapeutic effect targets of the treatment.

The design method. Strategies to modify and construct biological systems having desired properties can be devised based on definite design principles and simulations, instead of blind trial-and-error.

As it was mentioned above, in reality analysis of system dynamics and understanding the system structure are overlapping processes. In some cases analysis of the system dynamics can give useful predictions in system structure (new interactions, additional member of system). Different methods can be used to study the dynamical properties of the system:

• analysis of steady-states allows to find the systems states when there are no dynamical changes in system components.

• stability and sensitivity analyses provide insights into how systems behavior changes when stimuli and rate constants are modified to reflect dynamic behavior.

• bifurcation analysis, in which a dynamic simulator is coupled with analysis tools, can provide a detailed illustration of dynamic behavior.[74, 75]

• flux balance analysis[76] can be used to predict the different metabolic patterns as it was done, for instance, in Ref. [77] for predicting the switching in of the metabolic pathways in *Escherichia coli* under different nutritional conditions based on knowledge of only the metabolic network structure.

The choice of the analytical methods depends on availability of the data that can be incorporated into the model and the nature of the model.

It is important to know the main properties of the complex system under investigation, such as robustness.

Robustness is a central issue in all complex systems and it is very essential for understanding of the biological object functioning at the system level. Robust behavior in biochemical networks has been reported long time ago in Refs. [78, 79] as well as in more recent papers.[80–83] Robustness can be defined as the preservation of particular characteristics despite uncertainty in components or the environment.[84] Robust systems exhibit the following phenomenological properties:[64]

• *adaptation*, which denotes the ability to cope with environmental changes;

• *parameter insensitivity*, which indicates a system's relative insensitivity (to a certain extent) to specific kinetic parameters;

• *graceful degradation*, which reflects the characteristic slow degradation of a system's functions after damage, rather than catastrophic failure.

All the above features are present in many of the CI methods and techniques and make them very suitable to modeling complex biological systems.

The main feature of the evolutionary biology, the converged evolution, is that it leads to nearly optimal systems with similar gross characteristics, so simple arguments based on optimal design can explain functional relations between variables across many scales.[85, 86] Three other key elements (discovered with the use of computational modeling and experimentation) of the organizational principles used by cells are noted in Refs. [87–89]:

• Ultrasensivity, a response that is more sensitive to ligand concentration as compared to standard responses defined by the Michaelis-Menten equations;[90–92]

• Multistability, an existence of 2 or more stable state for the regulating network[93, 94]

• Rhythmic behavior, functioning as a systemic oscillator. In Ref. [95] the gene regulatory network with this property was described: three transcriptional repressors were used to build oscillating network in *Escherichia coli*.

More general principles which seem to be necessary for the operation of a living system (and peculiar to the complex biological systems) were presented in Ref. [96]:

• Program, plan describing ingredients and interactions between them as living system persist through time.

• Improvisation, the ability to change the program with respect to changes in environment.

• Compartmentalization, division of the organisms on smaller compartments in order to centralize and specialize certain functions.

• Energy, living organism is open system metabolizing energy.

• Regeneration, resynthesis of the constituents of the system.

• Adaptability, fast response that allow survival in quickly changing environments.

• Seclusion, the ability to allow thousands of reactions to occur with the high efficiency in the tiny volume of living cells.

Revealing all these characteristics of a complex living system helps choosing an appropriate method for their modeling, and also constitutes an inspiration for the development of new CI methods that posses these features.

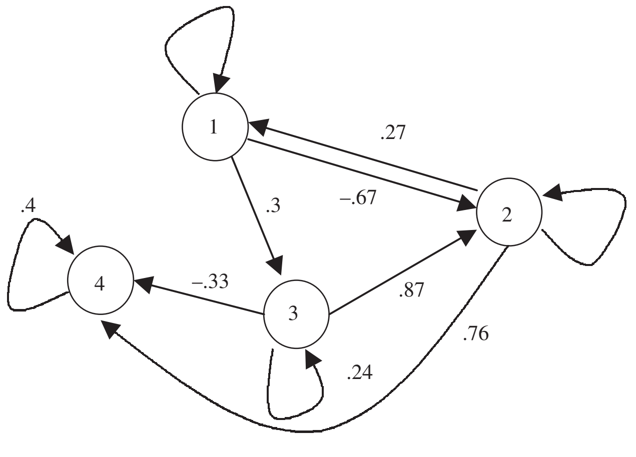


Fig. 7. A simple gene regulatory network (GRN) representing only 4 genes (the nodes) and their relative interaction strength—the arcs. Functions are used to calculate the activity of each gene depending on the activity of other genes in the network, which functions are not shown.

Modeling living cells *in silico* (in a computer) has many implications, one of them is testing new drugs through simulation rather than on patients. According to Ref. [97] human trials fail for 70–75% of the drugs that enter them.

Modeling gene regulatory networks (GRN) is the task of creating a dynamic interaction network between genes that defines the next time expression of genes based on their previous levels of expression. A simple GRN of 4 genes is shown in Figure 7.

A detailed discussion of the methods for GRN modeling can be found in Refs. [89, 98–104]. Models of GRN, derived from gene expression RNA data, have been developed with the use of different mathematical and computational methods, such as: statistical correlation techniques;[105, 106] evolutionary computation;[107, 108] ANN;[109, 110] differential equations, both ordinary and partial;[111] Boolean models (example is given in Fig. 8); kinetic models (example is given in Fig. 9); State-based models (see Section 2) and others.[102, 112] In Ref. [113] a simple GRN model of 5 genes is derived from time course gene expression data of leukemia cell lines U937 treated with retinoic acid with two phenotype states—positive and negative. The model uses ECOS.[10]

Despite of the variety of different methods used so far for modeling GRN and for systems biology in general, there is no a single method that will suit all requirements to model a complex biological system, especially to meet

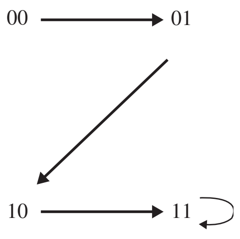


Fig. 8. Diagram representation of a Boolean network for a set of two genes.

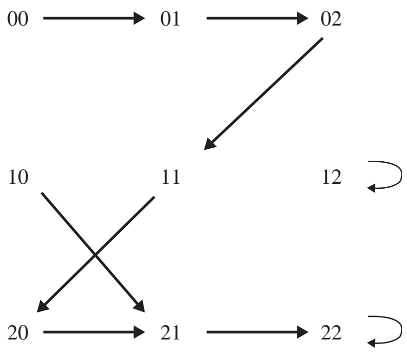


Fig. 9. Diagram representation of a kinetic logic model for a set of two genes and three states of gene expressions. The system has two stable states ("12," "22").

the requirements for adaptation, robustness, information integration. Novel methods of CI are needed, that include methods for model integration.

4. DISCUSSIONS: IMPLICATIONS FOR MEDICINE AND NANOTECHNOLOGY

The results obtained through the application of the CB and BI methods to biological problems, including the methods of CI, have a tremendous impact on the development of new drugs and treatments in Medicine on the one hand, and on the development of new computational methods and techniques on the other hand.

Profiling gene and protein expression using DNA and protein arrays has a tremendous impact in molecular-based classifications of diseases. There are two important tasks among others in this area: finding the correlation between subsets of genes/proteins and disease features (progression, localization, etc.); identifying the smallest informative set of genes/proteins associated with specific disease features (see Fig. 6).

The microarray technology offers an opportunity to screen thousands of genes simultaneously to be monitored in parallel. New disease subtypes or molecular distinct forms of the disease can be identified with the use of this technology: B-cell lymphoma,[114] two molecularly distinct forms of diffuse large B-cell lymphoma with gene expression patterns indicative of different stages of B-cell differentiation; breast tumors,[115, 116] gene expression patterns provided a distinctive molecular portrait of each tumor in a set of 65 surgical specimens of human breast tumors from 42 different individuals; human acute leukemia,[117] automatic discovery the distinction between acute myeloid leukemia and acute lymphoblastic leukemia. In Ref. [118] cDNA microarray technology was used to explore variation in gene expression in 60 cell lines of human cancer and consistent relationship between the gene expression patterns and the tissue of origin was found. Specific features of these gene expression patterns appeared to be related top physiological properties of the cell lines (doubling time in culture, drug metabolism, interferon response).

For some cases DNA microarray technology is inadequate method for studying, as it was noted in Ref. [119] for autoimmune diseases:

• disease may not manifest at the RNA level, but rather at protein one;

• protein function can be regulated by posttranslational modifications.

• nonpredictive correlations between RNA expression and protein expression and function.

The correlation between levels of mRNA measured in oligonucleotide microarrays and protein is important issue in DNA microarray technology. The lack of this correlation means that predictive property of the gene expression is independent of gene function. There is no strict linear relationship between genes and the 'proteome' of a cell. Proteomics is complementary to genomics because it focuses on the gene products and for this reason proteomics directly contributes to drug development as almost all drugs are directed against proteins.

Proteomics is a promising approach to the: identification of new diagnostic tools (identification of disease markers or proteins that appear or disappear during the course of a disease), development of drugs, improvement of efficiency of clinical trials (availability of biologically relevant markers for drug efficacy and safety); clinical diagnostic testing.

These approaches include: the analysis of protein expression in normal and disease tissue, analysis of secreted proteins in cell lines and primary cultures, direct serum protein profiling. Aberrantly expressed proteins might represent new markers. Mass spectrometry allows yielding comprehensive profiles of peptides and proteins without the need of first separate them and it is highly suited for marker identification.

The changes in protein expression that enable tumor to initiate and progress in the local tissue microenvironment were analyzed in with the use of antibody microarray. It was demonstrated that quantitative, and potentially qualitative, differences in expression patterns of multiple proteins within epithelial cells reproducibly correlate with tumor progression.

A reverse-phase protein array approach with immobilization of tissue's proteins has been reported in Ref. [120]. These arrays were used for screening of molecular markers and pathway targets in patient matched human tissue during disease progression. In contrast to previous protein arrays that immobilize the probe, reverse phase protein arrays immobilize the whole repertoire of patient proteins that represent the state of individual tissue cell populations undergoing disease transitions. A high degree of sensitivity, precision and linearity was achieved, making it possible to quantify the phosphorylated status of signal proteins in human tissue cell subpopulations.

Acknowledgments: The authors thank the editor-in-chief of the journal for his invitation and for his further encouragement in the preparation of this paper.

References

1. L. A. Zadeh, *Information and Control* 8, 338 (**1965**).
2. J. C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum Press, New York (**1981**).
3. V. Vapnik, Statistical Learning Theory, John Wiley and Sons, Inc. (**1998**).
4. S. Amari, *Proc. IEEE* 78, 1143 (**1990**).
5. C. Bishop, Neural Networks for Pattern Recognition, Oxford University Press (**1995**).
6. T. Kohonen, Self-Organizing Maps, Springer Verlag (**1997**).
7. S. Amari and N. Kasabov, Brain-like Computing and Intelligent Information Systems, Springer Verlag (**1997**).
8. M. Arbib, The Handbook of Brain Theory and Neural Networks, MIT Press (**2003**).
9. N. Kasabov, Foundations of Neural Networks, Fuzzy Systems and Knowledge Engineering, MIT Press (**1996**).
10. N. Kasabov, *Evolving Connectionist Systems—Methods and Applications in Bioinformatics, Brain Study and Intelligent Machines*, Springer Verlag, London, New York (**2002**).
11. D. E. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley, Reading, MA (**1989**).
12. J. D. Watson and F. H. C. Crick, *Nature* 171, 737 (**1953**).
13. B. A. Shapiro and W. Kasprzak, *J. Molecular Graphics* 14, 194 (**1996**).
14. B. A. Shapiro, W. Kasprzak, J. C. Wu, and K. Currey, in Pattern Discovery in Biomolecular Data, edited by J. T. L. Wang, B. A. Shapiro, and D. Shasha, Oxford University Press, Oxford, New York (**1999**), p. 183.
15. A. J. Gibbs and G. A. McIntyre, *Eur. J. Biochem.* 16, 1 (**1970**).
16. J. V. J. Maizel and R. P. Lenk, *Proc. Natl. Acad. Sci. USA* 78, 7665 (**1981**).
17. D. J. States and M. S. Boguski, in Sequence Analysis Primer, edited by M. Gribskov and J. Devereux, Stockton Press, New York (**1991**), p. 92.
18. G. Grillo, F. Licciulli, S. Liuni, E. Sbisa, and G. Pesole, *Nucleic Acids Res.* 31, 3608 (**2003**).
19. V. Matys, E. Fricke, R. Geffers, E. Gossling, M. Haubrock, R. Hehl, K. Hornischer, D. Karas, A. E. Kel, O. V. Kel-Margoulis, D. U. Kloos, S. Land, B. Lewicki-Potapov, H. Michael, R. Munch, I. Reuter, S. Rotert, H. Saxel, M. Scheer, S. Thiele, and E. Wingender, *Nucleic Acids Res.* 31, 374 (**2003**).
20. K. Quandt, K. Frech, H. Karas, E. Wingender, and T. Werner, *Nucleic Acids Res.* 23, 4878 (**1995**).
21. S. F. Altshul, *J. Mol. Biol.* 219, 555 (**1991**).
22. D. J. States, W. Gish, and S. F. Altshul, *Methods* 3, 66 (**1991**).
23. S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, *J. Mol. Biol.* 215, 403 (**1990**).
24. S. F. Altschul, T. L. Madden, A. A. Schaffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman, *Nucleic Acids Res.* 25, 3389 (**1997**).
25. V. C. Wasinger, S. J. Cordwell, A. Cerpa-Poljak, J. X. Yan, A. A. Gooley, M. R. Wilkins, M. W. Duncan, R. Harris, K. L. Williams, and I. Humphery-Smith, *Electrophoresis* 16, 1090 (**1995**).
26. J. S. Richardson, *Biophysics* 63, 1186 (**1992**).
27. J. L. McClelland and D. E. Rummelhart, in Explorations in Parallel Distributed Processing, MIT Press, Cambridge, MA (**1988**), Vol. 3, p. 318.
28. D. G. Kneller, F. E. Cohen, and R. Langridge, *J. Mol. Biol.* 214, 171 (**1990**).
29. B. Rost and C. Sander, *Proc. Nat. Acad. Sci.* 90, 7558 (**1993**).
30. B. Rost and C. Sander, *J. Mol. Biol.* 232, 584 (**1993**).
31. B. Rost and C. Sander, *Protein* 19, 55 (**1994**).
32. J. A. Cuff and G. J. Barton, *Proteins* 34, 508 (**1999**).
33. N. Qian and T. Sejnowski, *J. Theor. Biol.* 202, 865 (**1988**).
34. N. Guex and M. C. Peitsch, *Electrophoresis* 18, 2714 (**1997**).
35. M. C. Peitsch, *Biochem. Soc. Trans.* 24, 274 (**1996**).

36. O. Lund, K. Frimand, J. Gorodkin, H. Bohr, J. Bohr, J. Hansen, and S. Brunak, *Protein Eng.* 10, 1241 **(1997)**.
37. P. A. Bates, L. A. Kelley, R. M. MaxCallum, and M. J. E. Sternberg, *Proteins: Structure, Function and Genetic, Suppl.* 5, 39 **(2001)**.
38. P. A. Bates and M. J. E. Sternberg, *Proteins: Structure, Function and Genetic, Suppl.* 3, 47 **(1999)**.
39. B. Contreras-Moreira and P. A. Bates, *Bioinformatics* 18, 1141 **(2002)**.
40. I. N. Shindyalov and P. E. Bourne, *Forth Meeting on the Critical Assessment of Techniques for Protein Structure Prediction* **(2000)**, p. A-92.
41. I. N. Shindyalov and P. E. Bourne, *Nucleic Acids Res.* 29, 228 **(2001)**.
42. D. N. Chakravarti, B. Chakravarti, and I. Moutsatsos, *Comp. Proteomics Suppl.* 32, S4 **(2002)**.
43. R. C. Beavis and D. Fenyo, in Proteomics: A Trends Guide, edited by W. Blackstock and M. Mann, Elsevier, Amsterdam and New York **(2000)**, p. 22.
44. D. Fenyo, *Curr. Opin. Biotechnol.* 11, 391 **(2000)**.
45. N. L. Anderson, J. Taylor, A. E. Scandora, B. P. Coulter, and N. G. Anderson, *Clin. Chem.* 27, 1807 **(1981)**.
46. J. I. Garrels, *J. Biol. Chem.* 254, 7961 **(1979)**.
47. P. F. Lemkin and L. E. Lipkin, *Comput. Biomed. Res.* 14, 272 **(1981)**.
48. R. Appel, D. Hochstrasser, C. Roch, M. Funk, A. F. Muller, and C. Pellegrini, *Electrophoresis* 9, 136 **(1988)**.
49. R. D. Appel, D. F. Hochstrasser, M. Funk, J. R. Vargas, C. Pelegrini, A. F. Muller, and J. R. Scherrer, *Electrophoresis* 12, 722 **(1991)**.
50. T. Pun, D. F. Hochstrasser, R. D. Appel, M. Funk, V. Villars-Augsburger, and C. Pelegrini, *Appl. Theor. Electrophor.* 1, 3 **(1988)**.
51. D. G. Rowlands, A. Flook, P. I. Payne, A. van Hoff, T. Niblett, and S. McKee, *Electrophoresis* 9, 820 **(1988)**.
52. J. I. Garrels, *J. Biol. Chem.* 264, 5269 **(1989)**.
53. W. J. Browne, A. C. North, D. C. Phillips, K. Brew, T. C. Vanaman, and R. L. Hill, *J. Mol. Biol.* 42, 65 **(1969)**.
54. J. Greer, *Proc. Natl. Acad. Sci. USA* 77, 3393 **(1980)**.
55. J. Greer, *Proteins* 7, 317 **(1990)**.
56. R. Sanchez, U. Pieper, F. Melo, N. Eswar, M. A. Marti-Renom, M. S. Madhusudhan, N. Mirkovic, and A. Sali, *Nat. Struct. Biol.* 7, 986 **(2000)**.
57. E. Krieger, S. B. Nabuurs, and G. Vriend, *Methods Biochem. Anal.* 44, 509 **(2003)**.
58. A. Levchenko, *Mol. Biol. Rep.* 28, 83 **(2001)**.
59. A. J. Lotka, *J. Am. Chem. Soc.* 42, 1595 **(1920)**.
60. V. Volterra, *Mem. Acad. Lincei.* 2, 31 **(1926)**.
61. A. L. Hodgkin and A. F. Huxley, *J. Physiol.* 117, 500 **(1952)**.
62. M. Eigen, *Naturwissenschaften* 58, 465 **(1971)**.
63. A. Gierer and H. Meinhardt, *Kybernetik* 12, 30 **(1972)**.
64. H. Kitano, *Nature* 420, 206 **(2002)**.
65. M. Futschik and N. Kasabov, in *RECOMB'2001 Proceedings—Currents in Computational Molecular Biology*, edited by T. Lengauer and D. Sankoff, Montreal **(2001)**, p. 175.
66. J. Khan, J. Wei, M. Ringner, L. H. Saal, M. Ladanyi, F. Westerman, F. Berthold, M. Schwab, C. R. A. Nionescu, C. Peterson, and P. S. M. Meltzer, *Nature Med.* 7, 673 **(2001)**.
67. J. Gibbs, *Science* 287, 1969 **(2000)**.
68. C. Sander, *Science* 287, 1977 **(2000)**.
69. D. Noble, *Science* 295, 1678 **(2002)**.
70. N. Weiner, Cybernetics or Control and Communication in the Animal and the Machine, MIT Press, Cambridge, MA **(1948)**.
71. L. von Bertalnffy, Modern Theories of Development: An Introduction to Theoretical Biology, Oxford University Press, New York **(1933)**.
72. M. A. Savageau, Biochemical Systems Theory, Addison-Wesley, Reading, MA **(1976)**.
73. H. Kitano, (ed.), in Foundations of Systems Biology, MIT Press, Cambridge, MA **(2001)**.
74. K. C. Chen, A. Csikasz-Nagy, G. B. Val, B. Novak, and J. J. Tyson, *Mol. Biol. Cell* 11, 369 **(2000)**.
75. M. T. Borisuk and J. J. Tyson, *J. Theor. Biol.* 195, 69 **(1998)**.
76. K. J. Kauffman, P. Prakash, and J. S. Edwards, *Curr. Opin. Biotechnol.* 14, 491 **(2003)**.
77. J. S. Edwards, R. U. Ibarra, and P. B. O., *Nature Biotechnol.* 19, 125 **(2001)**.
78. M. A. Savageau, *Curr. Topics Cellular Regulation* 6, 63 **(1972)**.
79. H. Kacser and J. A. Burns, *Symp. Soc. Exp. Biol.* 27, 65 **(1973)**.
80. U. Alon, M. G. Surette, N. Barkai, and S. Leibler, *Nature* 397, 168 **(1999)**.
81. G. von Dassow, E. Mier, M. Munro, and M. Odell, *Nature* 406, 188 **(2000)**.
82. T.-M. Yi, Y. Huang, M. I. Simon, and J. Doyle, *Proc. Natl. Acad. Sci. USA* 97, 4649 **(2000)**.
83. H. Kurata and K. Taira, *Proc. Fourth Ann. Int. Conf. Comput. Mol. Biol.*, Tokyo, Japan **(2000)**, p. 36.
84. M. E. Csete and J. C. Doyle, *Science* 295, 1664 **(2002)**.
85. Scaling in Biology, Oxford University Press, New York **(2000)**.
86. J. Whitfield, *Nature* 413, 342 **(2001)**.
87. S. R. Neves and R. Iyengar, *BioEssays* 24, 1110 **(2002)**.
88. P. Smolen, D. A. Baxter, and J. H. Byrne, *AJP—Cell Physiology* 274, C531 **(1998)**.
89. H. Jeff, I. Farren, D. Milos, M. David, and J. J. Collins, *Chaos: An Interdisciplinary Journal of Nonlinear Science* 11, 207 **(2001)**.
90. D. E. Koshland, Jr., A. Goldbeter, and J. B. Stock, *Science* 217, 220 **(1982)**.
91. A. Goldbeter and D. E. Koshland, Jr., *Proc. Natl. Acad. Sci. USA* 78, 6840 **(1981)**.
92. J. E. Ferrell, Jr. and E. M. Machleder, *Science* 280, 895 **(1998)**.
93. James E. Ferrell and X. Wen, *Chaos: An Interdisciplinary Journal of Nonlinear Science* 11, 227 **(2001)**.
94. Upinder S. Bhalla and I. Ravi, *Chaos: An Interdisciplinary Journal of Nonlinear Science* 11, 221 **(2001)**.
95. M. B. Elowitz and S. Leibler, *Nature* 403, 335 **(2000)**.
96. J. Koshland, *Science* 295, 2215 **(2002)**.
97. R. Zacks, *MIT Technol. Rev.* 37 **(2001)**.
98. J. Hasty, D. McMillen, I. Farren, and J. J. Collins, *Nature Rev. Genet.* 2, 268 **(2001)**.
99. G. Weng, U. S. Bhalla, and R. Iyengar, *Science* 284, 92 **(1999)**.
100. P. Smolen, D. A. Baxter, and J. H. Byrne, *Neuron* 26, 567 **(2000)**.
101. U. Bhalla and Ravi Iyengar, *Science* 283, 381 **(1999)**.
102. H. de Jong, *J. Comput. Biol.* 9, 67 **(2002)**.
103. S. Huang, *Pharmacogenomics* 2, 203 **(2001)**.
104. H. Bolouri and E. H. Davidson, *BioEssays* 24, 1118 **(2002)**.
105. P. D'Haeseleer, S. Liang, and R. Somogyi, *Bioinformatics* 16, 707 **(2000)**.
106. S. Liang, S. Fuhrman, and R. Somogyi, *Pacific Symposium on Biocomputing* 3, 18 **(1998)**.
107. S. Ando, E. Sakamoto, and H. Iba, *Proc. 6th Joint Conference on Information Sciences* **(2002)**, p. 1249.
108. G. Fogel and D. Corne, (eds.), Evolutionary Computation for Bioinformatics, Morgan Kaufmann Publication **(2003)**.
109. J. Vohradsky, *J. Biol. Chem.* 276, 361 **(2001)**.
110. J. Vohradsky, *The FASEB J.* 15, 846 **(2001)**.
111. K. W. Kohn and D. S. Dimitrov, Mathematical models of cell cycles. Computer Modeling and Simulation of Complex Biological Systems. edited by S. S. Iyengar, Boca Raton, CRC Press **(1998)**, pp. 101–123.
112. J. Bower and H. E. Bolouri, (eds.), Computational Modeling of Genetic and Biochemical Networks, The MIT Press, Cambridge, MA **(2001)**.
113. N. K. Kasabov, Z. S. H. Chan, V. Jain, I. Sidorov, and D. S. Dimitrov, Gene regulatory network discovery from time-series gene

expression data—A computational intelligence approach. *Lecture Notes in Computer Science* 3316, 1344 (**2004**).

114. A. A. Alizadeh, M. B. Eisen, R. E. Davis, C. Ma, I. S. Lossos, A. Rosenwald, J. C. Boldrick, H. Sabet, T. Tran, X. Yu, J. I. Powell, L. Yang, G. E. Marti, T. Moore, J. J. Hudson, L. Lu, D. B. Lewis, R. Tibshirani, G. Sherlock, W. C. Chan, T. C. Greiner, D. D. Weisenburger, J. O. Armitage, R. Warnke, and L. M. Staudt, *Nature* 403, 503 (**2000**).

115. C. M. Perou, T. Sorlie, M. B. Eisen, M. van de Rijn, S. Jeffrey, C. A. Rees, J. R. Pollack, D. T. Ross, H. Jonsen, L. A. Akslen, O. Fluge, A. Pergamenshikov, C. Williams, S. X. Zhu, P. E. Lonning, A.-L. Borresen-Dale, P. O. Brown, and D. Botstein, *Nature* 406, 747 (**2000**).

116. T. Sorlie, C. M. Perou, R. Tibshirani, T. Aas, S. Geisler, H. Johnsen, T. Hastie, M. B. Eisen, M. van de Rijn, S. S. Jeffrey, T. Thorsen, H. Quist, J. C. Matese, P. O. Brown, D. Botstein, P. E. Lonning, and A. L. Borresen-Dale, *Proc. Nat. Acad. Sci.* 98, 10869 (**2001**).

117. T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. L. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield, and E. S. Lander, *Science* 286, 531 (**1999**).

118. D. T. Ross, U. Scherf, M. B. Eisen, C. M. Perou, C. Rees, P. Spellman, V. Iyer, S. S. Jeffrey, R. M. Van de, M. Waltham, A. Pergamenschikov, J. C. Lee, D. Lashkari, D. Shalon, T. G. Myers, J. N. Weinstein, D. Botstein, and P. O. Brown, *Nat. Genet.* 24, 227 (**2000**).

119. W. H. Robinson, L. Steinman, and P. J. Utz, *Arthritis and Rheumatism* 46, 885 (**2002**).

120. C. P. Paweletz, L. Charboneau, V. E. Bichsel, N. L. Simone, T. Chen, J. W. Gillespie, M. R. Emmert-Buck, M. J. Roth, E. F. Petricoin, and L. A. Liotta, *Oncogene* 20, 1981 (**2001**).

Received: 12 October 2005. Accepted: 25 October 2005.